In the Claims

1. (Currently Amended) A processor comprising:

a plurality of thread partitionable resources that are each partitionable between a plurality of threads including a first thread and at least one other thread;

logic to receive a program instruction from a first thread directing said processor to suspend execution of said first thread ~~of said plurality of threads~~, and in response to said program instruction to cause the processor to suspend execution of the first thread and to relinquish portions of said plurality of thread partitionable resources associated with the first thread for use by other ones of said plurality of threads.

2. (Currently Amended) The processor of claim 1 wherein the program instruction is a suspend instruction which consists of a suspend opcode which explicitly directs the processor to suspend execution of the first thread and to relinquish portions of said plurality of thread partitionable resources associated with the first thread for use by other ones of said plurality of threads.

3. (Previously Presented) The processor of claim 1 wherein said logic is to cause the processor to suspend the first thread for a selected amount of time.

4. (Previously Presented) The processor of claim 3 wherein said selected amount of time

is a fixed amount of time.

5. (Previously Presented) The processor of claim 3 wherein said processor is to execute instructions from a second thread while said first thread is suspended.

6. (Currently Amended) The processor of claim 3 wherein said selected amount of time is programmable by at least one technique chosen from a set consisting of:

providing an operand in conjunction with the program instruction;

blowing fuses to set the selected amount;

~~programming the selected amount in a storage location in advance of decoding the program instruction;~~

setting the selected amount in microcode.

7. (Previously Presented) The processor of claim 1 wherein said plurality of thread partitionable resources comprises:

an instruction queue;

a register pool.

8. (Previously Presented) The processor of claim 7 further comprising:

a plurality of shared resources, said plurality of shared resources comprising:

a plurality of execution units;

a cache;

a scheduler;

a plurality of duplicated resources, said plurality of duplicated resources comprising:

a plurality of processor state variables;

an instruction pointer;

register renaming logic.

9. (Previously Presented) The processor of claim 8 wherein said plurality of thread partitionable resources further comprises:

a plurality of re-order buffers;

a plurality of store buffer entries.

10. (Previously Presented) The processor of claim 1 wherein said logic is further to cause the processor to resume execution of said first thread in response to an event.

11. (Previously Presented) The processor of claim 3 wherein said logic is further to cause the processor to ignore events until said selected amount of time has elapsed.

12. (Previously Presented) The processor of claim 1 wherein said processor is embodied in digital format on a computer readable medium.

13. (Previously Presented) A method comprising:

receiving a first opcode in a first thread of execution;

suspending said first thread for a selected amount of time in response to said first opcode;

relinquishing a plurality of thread partitionable resources in response to said first opcode.

14. (Previously Presented) The method of claim 13 wherein relinquishing comprises:

annealing the plurality of thread partitionable resources to become larger structures usable by fewer threads.

15. (Previously Presented) The method of claim 14 wherein relinquishing said plurality of thread partitionable resources comprises:

relinquishing a partition of an instruction queue;

relinquishing a plurality of registers from a register pool.

16. (Previously Presented) The method of claim 15 wherein relinquishing said plurality of thread partitionable resources further comprises:

relinquishing a plurality of store buffer entries;

relinquishing a plurality of re-order buffer entries.

17. (Previously Presented) The method of claim 13 wherein said selected amount of time is programmable by at least one technique chosen from a set consisting of:

providing an operand in conjunction with the first opcode;

blowing fuses to set the selected amount of time;

programming the selected amount of time in a storage location in advance of decoding the program instruction;

setting the selected amount of time in microcode.

18. (Previously Presented) A system comprising:

a memory to store a plurality of program threads, including a first thread and a second thread, said first thread including a first instruction;

a processor coupled to said memory, said processor including a plurality of thread partitionable resources and a plurality of shared resources, said processor to execute instructions from said memory, said processor, in response to execution of said first instruction to suspend said first thread and to relinquish portions of said plurality of thread partitionable resources.

19. (Previously Presented) The system of claim 18 wherein said processor is to execute said second thread from said memory while said first thread is suspended.

20. (Currently amended) The system of claim 19 wherein said processor is to suspend execution of said first thread in response to said first instruction for a selected amount of time, said selected amount of time is chosen by at least one technique chosen from a set consisting of:

providing an operand in conjunction with the program instruction;

blowing fuses to set the selected amount of time~~;~~

~~programming the selected amount of time in a storage location in advance of decoding the program instruction;~~

~~setting the select amount of time in microcode~~.

21. (Previously Presented) The system of claim 18 wherein said plurality of thread partitionable resources comprises:

an instruction queue;

a register pool.

22. (Previously Presented) The system of claim 21 wherein said processor further comprises:

a plurality of shared resources, said plurality of shared resources comprising:

a plurality of execution units;

a cache;

a scheduler;

a plurality of duplicated resources, said plurality of duplicated resources comprising:

a plurality of processor state variables;

an instruction pointer;

register renaming logic.

23. (Previously Presented) The system of claim 22 wherein said plurality of thread partitionable resources further comprises:

a plurality of re-order buffers;

a plurality of store buffer entries;

24. (Previously Presented) An apparatus comprising:

means for receiving a first instruction from a first thread;

means for suspending said first thread in response to said first instruction;

means for relinquishing a plurality of partitions of a plurality of resources;

means for re-partitioning said plurality of resources after a selected amount of time.

25. (Previously Presented) The apparatus of claim 24 wherein said first instruction is a macro-instruction from a user-executable program.

26. (Previously Presented) The apparatus of claim 25 wherein said plurality of resources comprises a register pool and an instruction queue.